Audit, Standardisation et Refactorisation de code source

1. Mise en Situation

L'entreprise *BiblioTech* est en crise. Un prototype critique, gestion_biblio_source_final.html, a été livré dans un état catastrophique par un ancien employé. Le code est illisible, non sécurisé et impossible à maintenir.

Votre responsable technique vous confie la mission de **sauver ce projet**. Vous ne devez pas ajouter de nouvelles fonctionnalités, mais vous devez assainir la base de code existante pour qu'elle devienne professionnelle, sécuritaire et maintenable.

2. Objectifs

1. Définir les règles à suivre (Standards).
2. Identifier les erreurs actuelles (Revue).
3. Appliquer les correctifs pour produire une version propre (Refactoring).

3. Travail à réaliser

Partie A : Définition des Standards (Le "Contrat") (5 points)

Avant de toucher au code, rédigez un document d'une page : "Standards de Développement V1.0".

Définissez vos règles concernant :

- **Structure des fichiers**
- **Nommage**
- **Modernité JS :** (Ex: Interdiction de var, utilisation de let/const, Arrow functions).
- **Formatage**
- **Etc...**

Partie B : Rapport d'Audit (Le Diagnostic) (10 points)

Analysez le fichier source original. Sans le corriger tout de suite, listez dans un tableau les problèmes majeurs :

- Failles de sécurité critiques (ex: injection de code).

- Code mort ou commentaires inutiles.
- Logique complexe inutile.
- Violations des standards définis en Partie A.
- Etc…

Partie C : Refactoring (La Correction) (10 points)

C'est l'étape principale. Créez un nouveau projet propre et réécrivez/nettoyez le code original en respectant vos standards.

Exigences techniques obligatoires pour le code rendu :

1. **Architecture**
2. **Lisibilité**
3. **Nommage**
4. **Sécurité**
5. **Logique**
6. **Fonctionnalité :** L'application doit fonctionner exactement comme avant (Ajout, Liste, Suppression, Recherche), mais sans les bugs.

C'est une excellente addition. Cela transforme l'exercice d'une simple correction de code en une simulation réaliste de **sauvetage de projet en équipe**.

Voici la **Partie D** à ajouter à l'examen. Elle remplace ou complète la partie individuelle de refactoring.

Partie D : Collaboration et Gestion de Version (Travail d'Équipe) (10 points)

Contexte : Vous travaillez maintenant en groupe. Le code source "pourri" (gestion_biblio_source_final.html) est la seule version existante. Vous ne devez **jamais** modifier la branche main directement. Votre objectif est de séparer et nettoyer ce code en travaillant en parallèle sans tout casser.

Protocole Git à suivre :

Étape 1 : Initialisation du Dépôt (À faire par un seul étudiant)

1. Créez un nouveau dépôt **privé** sur GitHub

2. Invitez vos coéquipiers comme "Collaborator".
3. Ajoutez le fichier gestion_biblio_source_final.html (le code fourni par le prof) à la racine.
4. Faites un commit : feat: ajout du code legacy initial.
5. Poussez ce code sur la branche **main**.
6. **Sécurité :** Créez immédiatement une branche develop à partir de main. C'est la branche où vous fusionnerez vos travaux.

Étape 2 : Création des Branches de Fonctionnalité

Chaque étudiant doit cloner le dépôt sur sa machine et créer sa propre branche de travail à partir de develop (et non de main).

Étape 3 : Le Refactoring en Parallèle (Travail Simultané)

Étape 4 : La Fusion (Merge) et Résolution de Conflits

C'est ici que la communication est cruciale.

Étape 5 : Livraison Finale

1. L'un des étudiants crée une PR finale de develop vers main.
2. Vérifiez que le code sur main est propre et que l'application fonctionne.
3. Créez une **Release** (tag) nommé v1.0.0-clean.

---

4. Livrables à remettre

Vous devez déposer un fichier ZIP nommé contenant :

1. **Le Dossier "Projet_Final"** contenant le ou les nouveaux fichiers de code
2. **Le Rapport** contenant :
    - Vos Standards de développement.
    - Votre Tableau des anomalies trouvées dans la version originale.
3. Rendre le repository github public et fournir le lien.

Conseil pour l'étudiant :

Commencez par formater le code original (indentation automatique) pour réussir à le lire, puis attaquez les problèmes un par un. Ne tentez pas de tout réécrire de mémoire.